APPENDIX 1

LAPI_Nopoll_wait Subroutine

Purpose

LAPI_Nopoll_wait- Provides a counter value and a state vector corresponding to a list of destinations from which a response is expected to be updated once all the messages being waited for that will arrive have arrived.

Library

Availability Library (liblapi_r.a)

C Syntax

```
# include <lapi.h>

void LAPI_Nopoll_wait (hndl,  cntr_ptr, val, cur_cntr_val)
lapi_handle_t hndl;
lapi_cntr_t *cntr_ptr;
int val;
int *cur_cntr_val;
```

FORTRAN Syntax

```
include 'lapif.h'

int LAPI_NOPOLL_WAIT (HNDL, CNTR_PTR, VAL,  CUR_CNTR_VAL, IERROR)
INTEGER hndl;
INTEGER cntr_ptr;
INTEGER val;
INTEGER cur_cntr_val;
INTEGER ierror;
```

Parameters

hndl **IN** Handle to the LAPI instance

cntr_ptr* **IN/OUT Pointer to the lapi_cntr_t structure

val **IN** The relative counter value starting from 1 to be reached before returning.

cur_cntr_val* **IN Returns the current integer value of counter if not NULL.

ierror **OUT** Specifies a FORTRAN return code. It is always the last argument

Description

This function first checks if the state of counter is WAIT. It returns immediately with a return error of LAPI_ERR_MULTIPLE_WAITERS if it is already in a WAIT state. Then it checks if the counter values for *val* and **cntr_ptr* are the same and returns immediately if the value has been reached. Otherwise, it records in the *lapi_cntr_t (*cntr_ptr)* structure the value to be reached. There are two fields associated with the counter; **dest_list* and **dest_status* array of size *val* or **NULL**. **dest_list* and

dest_status* arrays record task IDs from which the calling thread is waiting for a response. These fields should only be set using **LAPI_Setcntr_wstatus. This function will immediately return with a return value of **LAPI_ERR_PURGED_TASK** if the destinations are purged. If **dest_list* is NULL then the behavior of **LAPI_Purge_totask** is to wake up all the threads that were sleeping in the **LAPI_Nopoll_wait.**

Note: To use this function you must have the *lib_vers* field of *lapi_info_t* in **LAPI_Init** set to **L2_LIB** or later.

Return Values

LAPI_SUCCESS Indicates the state was successfully reset.

LAPI_ERR_PURGED_TASK
Returned early due to **LAPI_Purge_totask** being called.

LAPI_ERR_MULTIPLE_WAITERS
More than one thread is waiting for the counter

LAPI_ERR_BAD_PARAMETER
A parameter passed in was not valid.

Location

/usr/lib/liblapi_r.a

Related Information

Subroutines: **LAPI_Purge_totask, LAPI_Resume_totask,**
LAPI_Setcntr_wstatus

LAPI_Setcntr_wstatus Subroutine

Purpose

LAPI_Setcntr_wstatus - Sets a counter to a specified value and sets the associate destination list array and destination status array to the counter.

Library

Availability Library (liblapi_r.a)

C Syntax

#include <lapi.h>

int LAPI_Setcntr_wstatus *(hndl, cntr, val, dest_list, dest_status)*
lapi_handle_t *hndl;*
lapi_cntr_t* *cntr;*
int *val;*
uint **dest_list;*
int **dest_status;*

FORTRAN Syntax

include 'lapif.h'

LAPI_SETCNTR_*WSTATUS* *(hndl, cntr, val, dest_list, dest_status, ierror)*
INTEGER *hndl;*
INTEGER *cntr;*
INTEGER *val;*
INTEGER *dest_list;*
INTEGER *dest_status;*
INTEGER *ierror;*

Parameters

hndl **IN** The handle that specifies the LAPI context

cntr **IN/OUT** Specifies the address of the counter to be set. This parameter cannot be NULL.

val **IN** Specifies the value the counter needs to be set to.

dest_list* **IN Specifies an array of destinations waiting for this counter update or NULL.

**dest_status*
IN/OUT Specifies an array of status (or NULL) corresponding to *dest_list.*

ierror **OUT** Specifies a FORTRAN return code. It is always the last argument.

Description

This function sets the *cntr* to the appropriate value. It returns with LAPI_ERR_BAD_PARAMETER if *dest_list* is not NULL and **dest_status* is. *dest_list* and **dest_status* record the status of a task from where the thread calling **LAPI_Nopoll_wait()** is waiting for a response. Status has the following format:

LAPI_MSG_AWAIT_RESP

Not received or purged.

LAPI_MSG_RECVD
Received.

LAPI_MSG_PURGED
Purged and not received.

LAPI_MSG_PURGED_RCVD
Purged and received.

LAPI_MSG_INVALID
Not valid; the task is already purged.

Note: This function should not be used when the parallel application is running under the POE/LL environment.

Return Values

LAPI_SUCCESS Indicates sucessful completion.

LAPI_ERR_BAD_PARAMETER
Indicates that a parameter was passed in that was not valid.

Location

/usr/lib/liblapi_r.a

Related Information

Subroutines: **LAPI_Getcntr, LAPI_Nopoll_wait, LAPI_Purge_totask, LAPI_Setcntr**